MICROPROCESSOR EVALUATION AND SELECTION FOR SPECIFIED APPLICATIONS

A Thesis Submitted
in Partial Fulfilment of the Requirements
for the Degree of

MASTER OF TECHNOLOGY

by

J. B. BHATTACHARJEE

to the

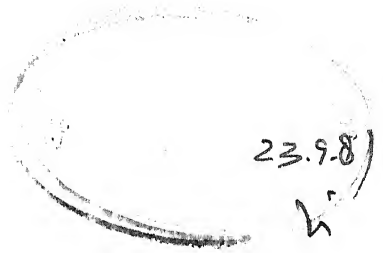
DEPARTMENT OF ELECTRICAL ENGINEERING
INDIAN INSTITUTE OF TECHNOLOGY KANPUR
SEPTEMBER, 1981

Dedicated to

Late Mrs. Meera Bhattacharjee

My Mother

EE-1981-M-BHA-MIC



CERTIFICATE

This is to certify that the work entitled MICROPROCESSOR EVALUATION AND SELECTION FOR SPECIFIED APPLICATIONS by J.B. Bhattacharjee has been carried out under our supervision and that this work has not been submitted elsewhere for a degree.

(V. Rajaraman)
Professor
Department of Electrical Engineering
and
Computer Sciences
Indian Institute of Technology
Kanpur.

(B. Srinivasan)
Lecturer
Department of Computer Sciences
Indian Institute of Technology
Kanpur.

ACKNOWLEDGEMENTS

I thank Dr. V. Rajaraman for suggesting this problem and for his guidance. He has been very understanding and his approach has been always humane.

I thank Dr. B. Srinivasan my second thesis supervisor.

I thank Dr. A.S. Sethi (Computer Science Dept.) for suggestions regarding the problem 'CRC Checking in Protocol' and Dr. P.R.K. Rao (Electrical Engineering Deptt.) for helping me with the problems of FFT and Electronic Telephone Exchange.

My dear friends, Santanu Datta, K.T. Sridhar and S. Ravindran have helped me during various stages of my thesis and to them I am grateful.

My thanks are also due to all my innumerable friends who have helped to make my stay at IIT Kanpur pleasant and memorable.

Finally, I would like to acknowledge the neat and careful typing of Shri C.M. Abraham.

J.B. Bhattacharjee

ABSTRACT

In this thesis a selection algorithm has been deviced by which an application engineer can select a microprocessor for his application. Such an automated procedure for selection is necessary, because, a large variety of microprocessors are available in the market today, and an application engineer has to go through all of them in detail, before deciding the suitability of a microprocessor, for his application, which is a time consuming task.

To tackle the problem, within the limits of realisation, four commonly used microprocessors viz., Intel 8085, Zilog Z-80, Motorola MC 6800 and Intel 8086 are chosen and are characterized. Also, a variety of applications have been studied and characterized. The selection algorithm tries to match, the parameters of the application with the characteristics of a microprocessor to find its suitability. Hence, an application engineer has only to characterize his application, and feed the data to selection algorithm to get the microprocessor which is best suited for his application.

CONTENTS

CHAPTER I

INTRODUCTION

1.0 PROLOGUE

One of the most notable developments in the last decade is the LSI technology. This technology currently dominates the microprocessor design and has greatly contributed to the fall in microprocessor cost. Consequently microprocessors have opened new areas of application which were earlier ignored due to non-availability and cost.

In the market, today, a large variety of microprocessors are available and this has been made possible with the LSI technology. An application engineer today is both bewildered and happy. Bewildered because there are many microprocessors with different characteristics to choose from and happy because, he knows that for his application atleast one of the microprocessors will be applicable. An application engineer has to go through in detail, all the microprocessors and analyse each one of them with his application in mind and then decide what is the best fit. Certainly it is a time consuming task when the varieties of microprocessors are large.

The aim of the thesis is to aid the application engineer in the selection process by automating it. To tackle the problem within limits of realization, four commonly used

microprocessors are chosen and tried to characterize them. Also we have tried to characterize a variety of applications. An algorithm is developed which, when specified with the characteristics of the application, tries to match the parameters of the application so specified, with the characteristics of the microprocessor /microprocessors. Hence, an application engineer has only to characterize his application and feed the data to this algorithm to get the microprocessor which is best suited for his application.

1.1 PREVIOUS WORK DONE

Microprocessor architecture is both different and diverse. Penney [Pen 77] gives the results of bench mark tests for several different microprocessors ranging from 4 bits to 16 bits. Programs corresponding to bench marks were written in the appropriate assembly language for each of the processors and the size of the code and speed of execution were calculated using the manufacturers data. These tests have shown that there is a definite improvement with word length and the execution times have most striking differences.

The bench mark tests considered by Penney are limited to the basic arithmetic operations. The execution times on a processor are dependent upon the number of operations and their relative frequency in an application. Hence, the comparison of microprocessors based on these execution times of the basic operations is not accurate.

Davis [Dav 79] has compared the architecture of three microprocessors. The comparative comments made by Davis are qualitative.

Farrar and Eidens [Far 80] have developed and evaluated analytically procedures for establishing microprocessor accuracy, computational capability and memory requirements for implementing linear quadratic Gaussian control logic. They have compared the arithmetic computation times obtained as function of parameters for Intel 8080 and Dec LSI-11/2. These procedures were evaluated by applying it to fifth order control logic of F 100 turbo engine model. The comparison of these two processors is restricted to only one application.

Our aim has been to design a generalised system for a variety of applications. Similarly, even though we have taken four microprocessors for our work, the algorithm we have developed can be used for all microprocessors by inserting the data about these new microprocessors. The developed algorithm tries to match the requirements of the application with that of the microprocessor for effective usage of the processor for the application.

1.2 OBJECTIVE OF THE THESIS

The objectives of the thesis can be stated as :

1) To study the characteristics of a class of commonly used microprocessors with particular reference to their

capabilities in I/O handling. The class includes Intel 8085, Motorola MC 6800, Zilog Z-80 and Intel 8086. The Intel 8086 is also in our study because it indicates the current trend towards higher word-length.

2) To develop a generalised model to characterize a group of applications which are I/O bound.

3) To evolve an algorithm which would enable a user to select a microprocessor to fit his application.

1.3 OUTLINE OF THE THESIS

In Chapter II, specification of the four microprocessors viz. Intel 8085, Motorola MC 6800, Zilog Z-80 and Intel 8086 are analysed. The factors which are important for various applications have been indicated. Tables have been drawn for quick reference of relative characteristics.

Various I/O handling techniques viz. software polling, vector interrupt approach and DMA, for all the four microprossors have been dealt with in detail in Chapter III.

The selected four microprocessors are characterized by different parameters and the parameters that are necessary for the selection algorithm are also identified in Chapter III.

In Chapter IV, a number of applications have been considered and the parameters which are critical for the application have been identified. Also inputs which will be used by the selection algorithm have been shown.

The selection algorithm is described in Chapter V. The results obtained by running the examples of Chapter IV are also discussed here. The results obtained theoretically and that obtained by the selection algorithm have been compared.

Chapter VI contains the conclusion and the suggestions for further extension of work.

CHAPTER II

STUDY OF MICROPROCESSOR SPECIFICATION

As has been indicated in the previous chapter, the microprocessor chosen were Intel 8085, Motorola MC 6800, Zilog Z-80 and Intel 8086. The rationale for the study of these microprocessors are :

1) 8085, 6800 and Z-80 are the most popular among the 8-bit microprocessors.
2) These have withstood the onslaught of time inspite of rapid changes in the hardware technology.
3) The characteristics of other 8-bit microprocessors are mostly found in the above class.
4) The Intel 8086, because it is a 16-bit microprocessor, which can also be used to simulate a 8-bit microprocessor. The software developed for the 8-bit 8085 can be used with minor modification in this microprocessor.
5) Intel 8086, indicates the current trend towards higher word length in microprocessors.

2.1 VARIOUS ASPECTS OF MICROPROCESSORS

Before going into the details of any particular microprocessor, it is worthwhile to study the general characteristics of a microprocessor. These can be classified into two broad categories.

1) Hardware

2) Software

Hardware can be further classified into :

i) Architecture

ii) Memory

iii) External logic requirements

iv) Speed

and the software can be split into :

i) Instruction repertoire

ii) I/O handling.

Architecture :

The main considerations are :

i) Number of primary accumulators and number of bits that it can handle.

ii) Number of secondary accumulators or general registers with number of bits that it can handle.

iii) Presence of index registers.

iv) Hardware stack with stack pointer.

v) Number of bits in the data bus.

vi) Number of bits in the address bus.

vii) Whether the data bus and address bus are multiplexed.

viii) Control lines.

ix) Duplication of Registers.

Memory :

Generally any 8-bit microprocessor can address about 64 K bytes of memory. So, the capacity of the memory addressing may not be an important factor. The important thing to look here are, whether the memory speed and the microprocessor speed match or not. If they do not match, then the execution of instruction will be delayed, which is undesirable.

Further, it may be necessary to use dynamic memory, in which case the memory has to be refreshed. Therefore, it is to be seen, whether microprocessor provides facilities for refreshing the dynamic memory.

External logic requirement :

When the microprocessor, is used as a system, then the number of chips (chip count) may increase, viz., some microprocessors do not have a built-in-logic for the clock, hence a separate chip has to be added for the clock logic with the microprocessor.

Speed :

Speed not only depends upon the cycle time of the microprocessor but also on the instruction execution time. If the cycle time is small, then the execution time will generally be small.

Instruction repertoire :

It is very important to go through the instruction repertoire of a microprocessor. This repertoire will convey the characteristics of the microprocessor. It is also important to note the number of machine cycles and/or clock cycles a particular instruction takes.

I/O handling :

A microprocessor by itself may be of little use, until it can interact with the environment. To react with environment, the I/O capability has to be looked into. In this thesis I/O handling has been a very important aspect and has been considered separately in Chapter III.

In addition to Hardware and Software aspects, there are two more points which have to be taken into consideration, while studying the suitability of a microprocessor for an application.

i) Cost : While deciding on a microprocessor the cost factor also has to be taken into account alongwith the other technical aspects.

ii) Familiarity : Some times, when a user is familiar with a microprocessor, he tends to use it even though it may not be the best for that application for that cost.

Of course, while dealing with applications of a microprocessors, the cost factor and the familiarity factor have not

been taken into account, since both are somewhat unscientific in nature.

2.2 FACTORS THAT INFLUENCE THE APPLICATION

It is not in the perview of this thesis to exhaustively explain each application and the factors that influence these application. A wide range of applications have been considered in detail in Chapter IV. Here, we discuss some of the application briefly.

i) Speed of the microprocessor and the execution of instructions: Consider an application, where inputing the data is not a critical factor, but their execution may be time bound, e.g., troposcattering application, speech synthesis, Here, the data is taken in and the processing is done on this data, and this processing is to be done within a certain period, though this period may be large. In these applications the speed of execution and the cycle time of the microprocessor are very important.

ii) Sampling speed of microprocessors : Here, the inputing of data is very important rather than processing. Since there are a number of input signals at different but reasonably high frequencies. In these applications the I/O characteristics of the microprocessor becomes a critical factor. The use of a microprocessor in telephone exchange is a pointer in this direction.

iii) Processing inside the memory : For some applications, it may be required that, the contents of the memory of a particular microprocessor has to be changed frequently. In which case, the microprocessor should be chosen having instructions, which directly affect the contents of the memory rather than going through the register. For this case, a microprocessor with an architecture allowing direct memory to memory operations are desirable.

iv) Mathematical processing : Here, the number of registers would be the determining factor and powerful arithmetic instructions are desirable.

The remaining chapter discusses the hardware and software features of the four microprocessors for our study.

2.3 INTEL-8085

It is an 8-bit microprocessor. Basically, it is an upward enhancement of Intel-8080. It has 16-bit address bus and 8-bit data bus. The lower 8-bit address bus is multiplexed with 8-bit data bus. The maximum and minimum cycle times are 2.0 μsecs. and 0.32 μsecs. respectively. The clock logic is built inside the 8085. It has one, 8-bit primary accumulator or simply accumulator. Six, 8-bit registers called B,C,D,E,H and L registers are available to the user. These could be used separately or in pairs. But the pairing can only be done in a particular fashion. The pairs are B-C, D-E and H-L. In addition to these,

there are two, 16-bit registers, program counter (not available to the user) and a stack pointer. Further, five one-bit registers indicate the various condition flags, viz., zero, sign, parity, carry and auxilliary carry. These set of five, one-bit registers are termed as PSW (processor status word). Internally (not accessible to the user) the 8085 has arithmetic, logic and control unit.

The input/output devices can be addressed using both memory mapped I/O and I/O mapped I/O scheme.

Additional features :

1) A general purpose interrupt request line.
2) Four hardware implemented vectoring line of which Trap has the highest priority. The other vectoring lines are termed RST7.5, RST6.5 and RST5.5 by INTEL and provides an unique address in memory for an interrupt. The Trap is non-maskable.

Instruction set :

Instructions could be one, two or three bytes long. The first byte always refers to the op-code. If the second and/or third byte are present, then they could be either an operand or an address of an operand.

Instructions may be divided into the following functional groups.

i) Data transfer group.

ii) Arithmetic group.

iii) Logical group.

iv) Branch group.

v) Stack, I/O group.

Addressing schemes :

Following are the addressing schemes available in this microprocessor.

i) Direct addressing : These are three bytes long instruction.

ii) Register addressing : This is a single byte instruction.

iii) Register indirect addressing : These are also single byte instruction.

iv) Immediate addressing : These could be two or three bytes long.

Memory and memory interface :

Since 16-bit parallel address is available, the microprocessor can address 64K bytes of memory address space directly.

The memory used should be compatible with the speed of microprocessor, otherwise 'Wait' cycles have to be introduced, if the memory used is slower compared to the microprocessor speed. Intel 8155 RAM can be interfaced with 8085 without any 'wait' cycles.

2.4 ZILOG Z-80

An 8-bit microprocessor. It has 16-bit address bus and 8-bit data bus. The maximum and the minimum cycle time are 11.21 μsecs. and 4.0 μsecs. respectively. The clock logic and the bus interface logic are built-in the Z-80. There is one 8-bit primary accumulator or simply an accumulator. Six, 8-bit registers have been provided. These six registers could be used separately or in pairs. The pairs are B-C, D-E and H-L. A processor status word is also present which gives the various condition flags, viz., carry, zero, sign, parity/overflow, auxilliary carry and subtract. These eight registers have been duplicated in the Z-80 itself. Two more, 8-bit registers are present and used as interrupt vector register and memory refresh register. Four 16-bit registers, viz., two index registers, stack pointer and program counter (not available to the user) are also provided.

Additional features :

1) The eight registers which have been duplicated, can be used for a single level interrupt without saving the registers in the memory. For single level interrupt, the processing is very fast.
2) A general purpose interrupt line. This can be used with the interrupt vector register as a hardware vectoring line.
3) A non-maskable interrupt line.

Instruction set :

i) Data transfer group.

ii) Arithmetic group.

iii) Logical group.

iv) Branch group.

v) Stack, I/O group.

vi) Block transfer facility.

Addressing schemes :

Same as that of 8085 with an additional advantage of Index registers. Since the index registers are of 16-bits, and the general registers are of 8-bits, it would be more appropriate to call these index registers as base registers. Also, individual bits of registers and memory could be set and tested.

Memory and memory interface :

Since this microprocessor is quite slow, there is no problem in finding a compatible memory.

2.5 MOTOROLA MC 6800

MC 6800 is an 8-bit microprocessor. Like the Z-80, this also has separate 16-bit address bus and 8-bit data bus. The machine cycle and the clock cycles are same. The bus interface logic is available inside this but not the clock logic. A separate clock logic has to be used with MC 6800. The maximum

and the minimum cycle times are 10.0 μsecs. and 1.0 μsec. respectively. There are two, 8-bit accumulators. These are both primary accumulators. The status register, which is again of 8-bits, indicates the various condition codes, viz., carry, overflow, sign, zero and auxilliary carry. Three 16-bit registers, program counter (not accessible by the user), Index register and stack pointer are available in this processor. Again the index register is of 16 bits (whereas the two accumulators are of 8 bits each) and therefore should be called a base register rather than index register.

The input/output devices can only be accessed by memory mapped I/O scheme.

Additional features :

1) A general purpose interrupt request line.
2) A non-maskable interrupt line.
3) Instruction set is very simple, even the control signals are very simple.

Instruction set :

There are no separate I/O instructions. The other groups are same as that of 8085.

Addressing schemes :

The addressing schemes are similar to that of Z-80.

Memory and memory interface unit :

Since it is slower compared to 8085, the compatibility of memory with MC 6800 is of no relevance, since memories with this speed are easily available.

2.6 INTEL 8086

It is a 16-bit microprocessor having the characteristics of both 8-bits as well as of 16 bits. The internal functioning of 8086 has been divided into two parts. Bus interface unit and the execution unit. These two can interact directly but quite often they perform different functions asynchronously. This improves the overall processor performance.

It contains four 16-bit general data registers that are also addresseble as eight, 8-bit registers, two, 16-bit memory base pointer registers and two,16-bit index registers. All data manipulation instructions apply to all registers but there are certain addressing modes which apply only to specific registers. A second bank of registers known as segment registers, which extends the chip's addressing capability are also incorporated in the 8086. A 16-bit status word is also available, out of that only 9 are used. They are carry, parity, auxilliary carry, zero, sign, trap, interrupt enable, direction and overflow. The address bus is of 20 bits, the data bus is of 16 bits. The maximum and the minimum cycle times are 2.0 μsecs. and 0.2 μsec. respectively.

Additional features :

1) 255 maskable vector interrupting could be utilised with 8086.

2) One non-maskable is also available.

3) Multiplication and division, instructions are also available.

Addressing schemes :

- Direct 16-bit offset address.
- Indirect through a base register, optionally with an 8-bit or 16-bit displacement.
- Indirect through the sum of one base register and one index register, optionally with an 8-bit or 16-bit displacement.

Instruction set :

Can be divided into the following functional groups :

- Data transfer group.
- Arithmetic group.
- Logic group.
- String manipulation.
- Control transfer.
- Process control.

Memory and memory interface :

The processor provides a 20-bit address to the memory. Therefore, one million bytes of memory could be addressed

directly. Physically the memory is organized in high and low bank having 512 K bytes each.

The memory has to be chosen to be compatible with this fast processor. Unfortunately, most of the supporting devices are not yet marketted.

Tables 2.1 and 2.2 provide a ready reference to various characteristics of the four microprocessors which were discussed in this chapter.

Table 2.1

Hardware Differences

	8085	6800	Z-80	8086
Minimum cycle time (μsecs)	0.32	1.0	4.0	0.2
Maximum cycle time (μsecs)	2.0	10.0	11.21	2.0
No. of primary accumulators	One	Two	One	-
No. of bits (accumulator)	8	8	8	-
No. of general registers	Six	-	Six	Eight
No. of bits (registers)	8	-	8	8
No. of Index registers	-	One	Two	Two
No. of bits (Index register)	-	16	16	16
General interrupt	One	One	One	-
Vector interrupt	Yes	Yes	Yes	Yes
No. of vector interrupt registers	-	-	One	-
Vector interrupt with support devices	Yes	Yes	Yes	Yes
Automatic saving of registers during interrupt	-	Yes	Yes	Yes
Addressing bits	16	16	16	20
Data bits	8	8	8	16
Multiplexing of the address and data bus	Yes	-	-	-
Memory refresh register	-	-	Yes	-
Duplication of registers	-	-	Yes	-

Table 2.2

Software Differences

	8085	6800	Z-80	8086
Bit manipulation	-	-	Yes	-
Memory bit manipulation	-	-	Yes	-
Block transfer instruction	-	-	Yes	Yes
Block search instruction	-	-	Yes	Yes
Multiplication instruction	-	-	-	Yes
Division instruction	-	-	-	Yes
Stack operations	Yes	Yes	Yes	Yes
Offset addressing	-	-	-	Yes
Decimal adjust add instruction	Yes	Yes	-	Yes
Direct addressing	Yes	Yes	Yes	Yes
Register addressing	Yes	Yes	Yes	Yes
Register indirect addressing	Yes	Yes	Yes	Yes
Immediate addressing	Yes	Yes	Yes	Yes
Indexed addressing	No	Yes	Yes	Yes

CHAPTER III

CHARACTERIZATION OF MICROPROCESSORS

3.0 PROLOGUE

A microprocessor needs to be interfaced with I/O devices for communicating with the environment. While buying a large computer system usually manufacturer provides the I/O devices and the interface for these I/O devices. The problem of selecting I/O devices, once the system has been selected may not be much of an ordeal, compared to the selection of the system itself.

But in a microprocessor environment, the user has the flexibility to choose I/O devices from a large number which are available in the market. In choosing the I/O devices for a microprocessor, the following consideration has to be taken into account.

1) Interfacing a slower I/O device to the processor.
2) Differences in format of the I/O to that accepted by the processor. The output from the I/O may be of 12 bits and that accepted by the processor may be 8 bits. Necessary interface has to be made in this case.
3) Electrical characteristics of the I/O and that of the processor may not be compatible.

Fortunately for the user, the microprocessor manufacturers have come up with a large number of interface units. To that extent the job of the user is simplified.

There are two ways by which a microprocessor can address the I/O.

1) Memory mapped I/O : In this scheme the address of I/O and that of memory are in one address space. Consequently, all the data transfer instructions of the microprocessor can be used for transferring data from and to memory and I/O devices. This is an added advantage to the software designer, as the complete set of memory reference instructions are available to him for data transfer. In this method, decoding of addresses becomes a problem. Decoder becomes complex and therefore costly.

2) I/O mapped I/O : According to this scheme the I/O devices and memory have distinct address spaces allocated to them. Hence, a microprocessor must have additional instructions for addressing the I/O devices. Extra control signals are also required to distinguish between memory and I/O.

3.1 DATA TRANSFER

I/O handling involves the following :

1) Data transfer itself.

2) Control signals.

Control signals are not considered, because mostly it is hardware oriented. We will talk about the data transfer part in some detail. Data transfer can be broadly classified in two categories.

- Programmed data transfer
- Direct memory access.

Programmed data transfer :

This can be further subdivided into three distinct methods.

(i) Synchronous transfer : Transfer of this kind are applicable when the speed of I/O device and that of microprocessor match. The user program may issue a command for a transfer of data. At the end of the execution of this command the data would have been transferred.

Suppose the speed of I/O is lower than that of the microprocessor in this case the microprocessor after getting the command from the user program, may give a ready signal to the I/O device, and during the time, the device takes to get ready, the microprocessor can perform some other task and as soon as the device gets ready, the microprocessor should be able to carry out the data transfer. Here, the problem is, that the user must know the precise time it takes for the I/O to get ready and only this time should be utilised by the microprocessor to do some other task, otherwise chaos will prevail.

(ii) Asynchronous transfer : The method is also known as handshaking. The data transfer can best be explained by the following Pascal like program.

```
Begin
   Issue instruction to the device to get ready;
   While (device not ready) do
      Begin
         Test device ready flag
      end;
   Issue data transfer instruction
end.
```

(iii) Interrupt driven data transfer : The main program and interrupt service subroutine are explained in Pascal like language below.

Main program :

```
   Begin
      While (interrupt not high) do
         Begin
            Fetch next instruction;
            Execute instruction
         end;
   Call interrupt service subroutine
   end.
```

Interrupt service subroutine :

```
Begin
    Save register contents;
    Save processor status word;
    Execute data transfer instruction;
    Restore register contents;
    Restore processor status word;
    Enable interrupt system
end;
```

This is the basic structure of the interrupt driven data transfer.

Direct memory access method :

In this method, data is directly transferred between I/O device and memory without going through the microprocessor registers. Again, this has been explained in a Pascal like language below.

```
Main program :
     ---
     ---
     ---
     ---
    While (input device not busy) do
       Begin
          While (input device ready) do
             Call DMA program
       end;
       ---
       ---
```

```
DMA program :
   Begin
      Initiate read command;
      Initiate data byte count;
      Initiate starting memory address;
      While (Request for data bus and address bus grant) do
         Begin
            While (count not zero) do
              Begin
                 Transfer data byte to memory;
                 Modify memory address to point to the next
                 location;
```

```
            Decrement count by one;
          end;
        Request release of data bus and address bus;
        Inform processor that transfer is complete
      end
    end;
```

The data transfers of the specified microprocessors namely Intel 8085, Motorola MC6800, Zilog Z-80 and Intel 8086 with their interface devices have been discussed in the following sections. The data transfer methods that have been considered are :

i) Software polling (in handshake mode)

ii) Interrupt driven data transfer.

iii) Direct memory access.

3.2 SOFTWARE POLLING FOR THE SPECIFIED MICROPROCESSORS

Intel 8085 : The interface unit chosen was program peripheral interface 8255. It has three ports A,B and C. Each is of 8 bits and bidirectional data transfer can be done through them. 8255 can be operated in three modes :

Mode 0 : Port A,B act as 8 bit input or output ports and the two halves of the port C, separately serve as 4 bit input or output ports.

Mode 1 : Port A,B may be used as input or output ports. Port C is used for control signals to and from the processor to the I/O device. The upper 4 bits of port C, acts as the control for the port A and the lower 4 bits act as controls for the port B. Therefore, this mode could be utilised for handshaking system.

Mode 2 : Port A acts as strobed bidirectional I/O bus. PC3-PC7 define the control signals for port A, Ports B and C may be used either in mode 0 and mode 1.

It is assumed that 8085 was used in conjunction with 8255 for the software polling and the following program was written to find out the, time required to transfer data from I/O device, in clock cycles.

Main program :	Clock cycles		Clock cycles
LOOP : IN(Port No)	10	RNZ INPUT 1	6
ANI(Number)	7	JMP LOOP	10
		Total	33

The main program has been written for only one input. If the number of inputs are more than one, then the first three instructions will be repeated for each input.

		Clock cycles		Clock cycles
INPUT 1 :	PUSH B	12	OUT(Port No)	10
	PUSH H	12	POP PSW	10
	PUSH D	12	POP D	10
	PUSH PSW	12	POP H	10
	LXI H,XXX	10	POP B	10
	IN (Port No)	10	RET	10
	MOV M,A	7		
	(Processing if any)		Total	132
	MOV A,M	7		

Total number of clock cycles required = 132 + 33 = 165.

MC 6800 : Peripheral interface adapter (PIA) 6820 along with MC 6800 constitute the software polling system for this microprocessor. The PIA 6820 has two ports A and B of 8 bits each. Each port has its own control register and data direction register. There are two lines provided for each port to facilitate the process of handshaking. The control register and data register can be accessed by the user program, so that necessary control bits may be written in these registers. Compared to Intel 8255, PIA 6820 operation is somewhat complicated.

Main Program :		Clock cycles		Clock cycles
LOOP :	LDA A (Port No)	3	BNE INPUT 1	4
	AND A,M	2	X : JMP LOOP	3
			Total	12

		Clock cycles		Clock cycles
INPUT 1 :	PSH B	4	STA Port No A	4
	PSH H	4	PUL PSW	4
	PSH B	4	PUL D	4
	PSH PSW	4	PUL H	4
	LDA A(Port No)	3	PUL B	4
	STA M,A	4	BRA X	4
	(Processing,if any)			
	LDA A,M	3	Total	50

Total number of clock cycles = 12 + 50 = 62.

Z-80 : Here also Zilog PIO has been used with Z-80.

Main program :		Clock cycles		Clock Cycles
LOOP :	IN (Port No)	10	JPNZ INPUT 1	10
	LD HL,No	10	X : JP LOOP	10
	AND A,HL	7	Total	47
INPUT 1	PUSH B	15	OUT (Port No)	11
	PUSH H	15	POP PSW	14
	PUSH D	15	POP D	14
	PUSH PSW	15	POP H	14
	LD H, Address	16	POP B	14
	IN (Port No)	10	JP X	10
	LD M,A	13	Total	189
	(Processing, if any)			
	LD A,M	13		

Intel 8086 : The peripheral interface chips are yet to be marketed for this microprocessor. It is assumed that they are present (with the similar characteristics as that of Intel 8225) and the following program was written.

Main program :	Clock cycles		Clock cycles
LOOP : IN (Port No)	10	X : JMP LOOP	15
AND AL,XXX	3		
JE INPUT 1	16	Total	44
INPUT 1 : PUSH AX	10	OUT (Port No)	10
PUSH BX	10	POP F	8
PUSH CX	10	POP DX	8
PUSH DX	10	POP CX	8
PUSH F	10	POP BX	8
IN (Port No)	10	POP AX	8
MOV Address,RA	9	JMP X	15
(Processing,if any)		Total	142
MOV RA,Address	8		

Total number of clock cycles = 44 + 142 = 186.

3.3 INTERRUPT DRIVEN DATA TRANSFER FOR SPECIFIED MICROPROCESSORS

Intel 8085 : Vectored interrupt takes less time compared to polled interrupt. It will be clear from the program written as follows. Hence, vectored interrupt has been considered for

the other cases too.

The interface used was programmed interrupt controller (PIC) 8259. The 8259 receives an interrupt from the I/O device, checks its priority and by itself interrupts the microprocessor if it is necessary.

PIC 8259, has to be initialized i.e. initialization command word and operation command word have to be loaded as required by the user.

Polled interrupt program :	Clock cycles		Clock cycles
JMP X	10	OUT(Port No)	10
Y : RET	10	POP PSW	10
X : PUSH B	12	POP D	10
PUSH H	12	POP H	10
PUSH D	12	POP B	10
PUSH PSW	12	JMP Y	10
LXI,H,XXX	7	To get the total time a call instruction has to be added	18
IN (Port No)	10		
MOV M,A	7		
(Processing,if any)		Total	177
MOV A,M	7		

3.4 VECTORED INTERRUPT

There are four lines available in 8085. These are TRAP, RST 7.5, RST 6.5 and RST 5.5. If any interrupt comes to any one of these lines, it jumps to specified locations. The table is given below.

Priority	Name	Location to which the program jumps
Highest	TRAP	24
1	RST 7.5	3C
2	RST 6.5	34
Least	RST 5.5	2C

The program for the vectored interrupt is given below.

	Clock cycles		Clock cycles
JMP X	10	OUT Port No	10
Y : RET	10	POP PSW	10
X : PUSH B	12	POP D	10
PUSH H	12	POP H	10
PUSH D	12	POP B	10
PUSH PSW	12	JMP Y	10
LXI, H,XXX	7	To get the total time another MOV instruction time has to be added since PC goes to stack	7
IN,Port No.	10		
MOV M,A	7		
(Processing,if any)		Total	166
MOV A,M	7		

Therefore, we see that the vectored interrupt is faster compared to the polled interrupt.

Zilog Z-80 : In this microprocessor, pop and push instructions are not required, when writing a program for vectored interrupt, since the processor itself takes care of it.

The interface unit compatible with Z-80 is Z-80 PIO.

Program :	Clock Cycles		Clock cycles
JP X	10	OUT Port No.	11
Y : RET	10	JP Y	10
RET I	14	One call instruction time has to added. For automatic pushing of registers are one more clock cycle has to be added	1
X : IN Post No.	10		
LD XXX, A	13		
(Processing, if any)		Total	109
LD A, XXX	13		

MC 6800 : The interface used with 6800 for vectored interrupt was 6828. While adding the total clock cycles, 7 clock cycles have to be added, because it takes 4 cycles to stack Ix and Iy 2 cycles for PC and 2 cycles for A and B.

Program	Clock cycles		Clock cycles
JMP X	3	STA A,I/O address	4
Y : RT I	10	JMP Y	3
X : LDA, A, I/O address	3	Stacking process	7
STA A, XXX	4		
(Processing, if any)		Total	37
LDA A,XXX	3		

Intel 8086 : Peripheral interface device has not been marketed yet. But all the same, a program has been written to find the time required for vectored interrupt process.

Program :	Clock cycles		Clock cycles
JMP X	15	OUTW [DX]	8
Y : IRET	32	JMP Y	15
LABEL 1 : Port No.			-----
LABEL 2 : XXX		Total	120
X : MOV DX,LABEL 1	14		
INW [DX]	8		
MOV LABEL 2, AX	14		
(Processing, if any)			
MOV AX, LABEL 2	14		

3.5 DMA FOR THE SPECIFIED MICROPROCESSORS

Intel 8085 : With 8085, 8257 is used as an interface for DMA operations. Address of the memory location to/from which data has to be dumped/taken out has to be loaded in 8257 by the user. Further, no of bytes to be transferred has also to be loaded into 8257. Hence, this time is to be added for calculation of total time. 8257 uses 4 clock cycles for transfer of one byte of data.

Program :	Clock cycles		Clock cycles
MVI A,	10	MVI A,	10
OUT DMA 1,	10	OUT	10
MVI A,	10	MVI A,	10
OUT DMA 1,	10	OUT	10
		For one byte of transfer	4
		Total	84

Z-80 : It has only one DMA channel, but that DMA channel can be used with interrupt, because there is a register in which interrupt vector address can be loaded.

Program :	Clock cycles		Clock cycles
LD A,Code	13	LD A,	13
OUT DMA,A	11	OUT DMA, A	11
LD A, Port No.	13	LD A,	13
OUT DMA, A	11	OUT DMA, A	11
LD A, Port No.	13	Data transfer	15
OUT DMA,A	11		
		Total	145

MC 6800 : This microprocessor uses 6844 as an interface device for the DMA transfer. There are four DMA channels in 6844. Channel control register and data chain control register of 6844 are to be initialized by the user.

Program :	Clock cycles		Clock cycles
LDA ACA, No.of bytes	2	LDA ACB, XXX	2
LDA ACB,No. of bytes	2	STA ACA, DMA addr	4
STA ACA, DMA byte reg.	4	STA ACB, DMA addr	4
STA ACB, DMA byte reg.	4	Transfer of data	2
LDA ACA, XXX	2		
		Total	26

Intel 8086 : No peripheral interface device is available. Hence, the program is not written for this.

Table 3.1 depicts in a nutshell the various characteristics of the microprocessor I/O handling with its interface unit.

3.6 SELECTION OF MICROPROCESSORS

The I/O handling of the four microprocessor have already been discussed and various programs have been written. This does give out the nature of a microprocessor. In real environment, the I/O handling capability is most important compared to the processing, though it would be unfair to disregard the processing capability of a microprocessor completely.

Now, let us look into the actual selection process based on I/O handling capability.

Software polling :

Intel 8085, MC 6800, Z-80 and Intel 8086 take 165, 62, 236 and 186 clock cycles respectively. From the data obtained

it is clear that MC 6800 is probably the best for software polling. But the question arises, can we take this value and use it for subsequent calculations? The answer is no. Because of the following reasons.

i) MC 6800 can operate only from 1.0 microseconds to 10.0 microseconds cycle time. Therefore, if a user wishes to use a microprocessor below 1.0 μsecs, the MC 6800 simply cannot be operated.

ii) 8086 can operate between 0.2 μsecs and 2.0 μsecs. 8085 can operate between 0.32 μsecs and 2.0 μsecs. Therefore, between 0.32 μsecs and 1.0 μsecs. both 8085 and 8086 could be used. Between 0.32 to 1.0 μsecs 8085 is preferred.

iii) Between 10.0 μsecs and 11.21 μsecs Z-80 could be used.

Vectored interrupt approach : Intel 8085, MC 6800 and Intel 8086 take 166, 109, 37 and 120 clock cycles.

1) Between 0.2 μsecs and 0.32 μsecs cycle time Intel 8086 is preferred.

2) Between 0.32 μsecs and 1.0 μsecs cycle time Intel 8085 is preferred.

3) From 1.0 μsecs to 10.0 μsecs. MC 6800 is preferred.

4) Between 10.0 and 11.21 μsecs Z-80 is used.

DMA approach :

1) Between 0.2 μsecs and 0.32 μsecs, nothing can be used, since 8086 peripheral interface for the DMA has not been marketed yet.
2) Between 0.32 μsecs and 1.0 μsecs cycle time 8085 is preferred. It requires 84 clock cycles.
3) Between 1.0 μsecs and 10.0 μsecs cycle time 6800 is preferred. It requires 26 clock cycles.
4) Between 10.0 and 11.21 μsecs. cycle time Z-80 is preferred. It requires 145 clock cycles.

A point that has not been emphasised as far as characterisation of a microprocessor for an application is that of cycle time. Every microprocessor can operate within a certain bandwidth of time. If this time is low, then the instruction execution time is also low (which covers both processing and I/O handling). Hence, cycle time is the main pivot used in deciding the usability of a microprocessor for an application in subsequent chapters. Table 3.2 indicates the differences in cycle time of the four microprocessors considered.

Table 3.1

Interface units and clock cycles required for I/O handling

	Intel 8085	MC 6800	Z-80	Intel 8086
Number of clock cycles required for software polling	165	62	236	186
Interface device used for the above	8255	PIA 6820	Zilog PIO	N/A
Number of clock cycles required for vector interrupt approach	166	37	109	120
Interface unit used for the above	PIC 8259	6828	Z-80 PIO	N/A
No. of clock cycles required for DMA (for one byte of data)	84	26	145	Cannot be computed since interface unit not available
Interface device used for above	8257	6844	Z-80 PIO	N/A

Table 3.2

Cycle time and I/O handling of the four microprocessors

I/O handling	Cycle time (in microseconds)			
	0.2 to 0.32	0.32 to 1.0	1.0 to 10.0	10.0 to 11.21
1. Microprocessor that can be used for software polling	8086	8085	6800	Z-80
2. Microprocessors that can be used for vector interrupt approach	8086	8085	6800	Z-80
3. Microprocessor that can be used for DMA approach	-	8085	6800	Z-80

CHAPTER IV

CHARACTERIZING APPLICATIONS

In this chapter applications from various fields are studied with respect to, using a microprocessor for controlling the process and the critical parameter values are identified. The application chosen for the study are : (1) Fast Fourier Transforms using Cooley-Tukey algorithm (FFT), (2) Microprocessor in fire control applications, (3) CRC checking in the protocol, (4) Microprocessor based spectral analysis for real time application, (5) Air data computer, (6) Microprocessor based electronic exchange.

Example 1 : Fast Fourier Transforms using Cooley-Tukey algorithm (FFT)

FFT is an algorithm which computes discrete fourier transforms of a sequence of data samples. With this technique the number of computations are reduced in finding the transforms.

The task of the microprocessor is to find the FFT from the following data.

1) Number of points for which FFT is to be developed. The optimum number of points which gives the true characteristics of inputs is 1024. But if the accuracy can be sacrificed then, number of points considered could be less [BHA 78].

2) The time required to do the above process = 620 milliseconds

3) The sampling time (i.e. of the input channel) = 10 samples every 6.2 milliseconds [10 Real + 10 imaginary].

Of course the third requirement mentioned above is quite superfluous once the second is considered. The time constraint was fixed to achieve the real time environment, since 10 samples (10 real and 10 imaginary) are taken every 6.2 milliseconds from the digital correlator of the troposcatter channel for measurements of Doppler effect in the troposcatter communication channel. For more details on this topic the reader is directed to refer to [BHA 78].

Computational time required

Number of complex multiplications required for this algorithm = $\frac{N}{2} \log_2 N = 5120$, where N = 1024.

Number of complex additions required = $N \log_2 N = 10240$.

Since one complex multiplication requires four real number multiplication and two real number additions, hence the total number of multiplications and additions required for the algorithm = 5120 x 4 + 10240x2 = 20480 + 20480. One 16x16 bit multiplication takes about 4 times the time taken by 8x8 bit multiplication. Therefore, number of multiplication to be done = 20480x4 = 81920. MC 6800, multiplication routine takes about 56 clock cycles [PEA 77].

Number of processing cycles required = 81920x56x8

= 36700160.0

1. The minimum cycle time for 8085 = 0.32×10^{-6} seconds.
   The total time taken for multiplication by 8085 = $(36700160.0)(0.32 \times 10^{-6})$ seconds = 8.74 seconds $>>$ 620 milliseconds.

2. The minimum cycle time for MC 6800 = 1×10^{-6} seconds.
   This cycle time is greater than that of Intel 8085. Hence, the time taken by 6800 will be much more than 620 milliseconds.

3. The minimum cycle time for Zilog Z-80 = 40×10^{-6} seconds.
   This cycle time is greater than that of Intel 8085. Hence this also will take much more time than the required limit of 6200 milliseconds.

Therefore, it is clear that Intel 8085, MC 6800 and Z-80 cannot be used for calculating FFT with this algorithm.

Let us take the case of 8086. It is a 16 bit microprocessor. The multiplication instruction takes 136 clock cycles. The minimum cycle time of 8086 is 0.2×10^{-6} seconds. Therefore, number of processing cycles required for multiplication by 8086 = 20480x136 = 2867200

The total time required for multiplication = $2867200 \times 0.2 \times 10^{-6}$

= 0.557056 seconds.

The add instruction takes 4 clock cycles.

Therefore, number of processing cycles required for addition

$= 20480 \times 4 = 81920.$

The time required for addition $= (81920.0)(0.2 \times 10^{-6}) =$

$= 0.008384$ seconds.

Therefore total processing cycles required for both multiplication and addition = 2867200.0.

The total time required for addition and multiplication

$= 0.56544$ seconds $<$ 620 milliseconds.

Hence 8086 could be used for the process theoretically. But actually it may not be so, because of I/O routines would be needed. We will take this up in chapter V. A graph has been drawn for the four microprocessors (Fig. 4.1) indicating the processing time requirements for different number of sample points.

In this particular example the emphasis has to be on the processing time. The critical factors are :

1. Cycle time of microprocessor :

   a. The processing has to be done within 620 milliseconds. Therefore the cycle time required to do this processing is = 0.01689 μsec. when we consider processing cycle = 36700160.0 out of the four microprocessors chosen, none of them can be operated at this cycle time. Hence none could be used.

b) Again the processing has to be done within 620 milliseconds. Therefore, the cycle time required to do this processing = 0.2162 μsec. when we consider the processing cycle = 2867200.0. Only 8086 is capable of operating at this cycle time.

2. Processing cycles : This gives the number of operations to be performed.

3. The sampling frequency.

Example 2 : Microprocessor in fire control application

The microprocessor used should be able to predict the future position of a target based on the present speed, direction and altitude of the target. It should also direct the fire to future target position such that the target and the fire reaches the predicted point simultaneously. For further details please refer to[ANA 77].

The sampling speed of the input = 100 milliseconds.

Number of clock cycles required to run the program = 135371.0 [ANA 77].

Let us consider the four microprocessors.

1. Intel 8086

The minimum cycle time = 0.2×10^{-6} seconds.

Therefore the total time required for running the program = $135371 \times 0.2 \times 10^{-6}$ second = 27.074 milliseconds $<$ 100 milliseconds.

2. Intel 8085

The minimum cycle time = 0.32×10^{-6} seconds.

Therefore the total time required for running the program = $135371 \times 0.32 \times 10^{-6}$ seconds

= 43.318 milliseconds $<$ 100 milliseconds.

3. MC 6800

The minimum cycle time = 1.0×10^{-6} seconds.

The total time required for running the program

= $135371 \times 1.0 \times 10^{-6}$ seconds

= 135.37 milliseconds $>$ 100 milliseconds.

4. Zilog Z-80

The minimum cycle time = 4.0×10^{-6} seconds.

The cycle time is more than that of MC 6800. Hence the time required for running the program, in this case also will be greater than 100 milliseconds. Therefore, theoretically, it is possible to use Intel 8085 and Intel 8086 for this purpose.

The inputs which are critical for this application are the following :

1. Cycle time : The cycle time required to complete the process is $\leq$ 0.73871 μsecs.
2. Processing cycles : Gives the number of operations to be performed is equal to 135371.
3. Sampling frequency.

Example 3 : CRC Checking in Protocols

In a network environment often the host computer has to take care of the line noises also. It has to implement the line protocol and also to take care of user's needs. Here we investigate the possibility of using a microprocessor to take away, a part of the load of a protocol implementation; in short decentralising the protocol activities.

Possibly the microprocessors can check for CRC (cyclic redundancy code) and also check whether packets are received in sequential order. Then the actual work the microprocessor is entrusted with the following :

1. Assembling the packets
2. Checking for CRC
3. Transmitting the message to the host.

Alternatively the microprocessor can fully handle the line protocol and just pass the message to the host. Analysing and investigating this possibility would be a time consuming task. We will now look at what the host has to do typically to implement a line protocols say one like DDCMP (digital data communication message protocol). DDCMP comprises various types of messages. Messages are broken into packets and sent on the line. Every message has a header followed by 16 bits of CRC and the data part if any with its CRC. Messages would be either data or control messages. Control messages have

only one header and are utilised for error free data transmission between two hosts. Control messages in DDCMP are the ACK, NAK, REP, STRT and STACK messages.

The header part of the data messages are 6 bytes long and have 2 bytes CRC appended to it. When a host receives a control message it checks for CRC and processes the information contained in the header. Some protocol variables that indicate the current state of protocol are modified and further request for the control message transmission are made if necessary.

ACK : Positive acknowledgement. When station B receives an error free message it informs station A of it through an ACK.

NAK : Negative acknowledgement. Station A can request information about the receipt of say the ith message and station B informs of any error or non-arrival of the ith message through a NAK.

REP : This is the request for information about the receipt of a particular message.

STRT and STACK : These messages are sent initially at stations start up - a start and start acknowledge respectively.

Positive acknowledgement are often Piggy backed on NAK or data messages.

Of these activites one could shift the assemblies of the message and checks for the CRC onto a microprocessor.

If the microprocessor has memory of its own to serve as buffer for messages (this is the present case) then the following operations can be carried out :

i) It assembles messages in its memory

ii) It checks for the CRC of the assembled messages

iii) It sends the messages with correct CRC on to the host machine. It also indicates the reception of a message.

The sampling rate of messages is 1200 bytes/sec. [0.833 milliseconds]. This is a standard value for data transfer in computer environment.

Using the known fast algorithm [WHI 75] a program has been written to accomplish the above indicated operations. The program is shown in Appendix A . The total number of clock cycles required to do the above three operations = 1206.0 [Refer Appendix A]. Here, again the processing part is critical and emphasised.

The critical points to be considered here are

1) Cycle time of microprocessor : The cycle time should be $\leq$ 0.69 μsec.

2) Processing cycles : 1206.0 operations are to be performed in a 833 milliseconds.

3) Sampling frequency.

Let us take the four microprocessors and calculate the processing time taken by each one of them.

1) Intel 8086 : Minimum cycle time = 0.2×10^{-6} sec.
Processing time = $(1206)(0.2 \times 10^{-6}) = 0.242$ milliseconds
$<$ 0.833 milliseconds.

2) Intel 8085 : Minimum cycle time = 0.32×10^{-6} sec.
Processing time = $(1206)(0.32 \times 10^{-6}) = 0.3859$ milliseconds
$<$ 0.8333 milliseconds.

3) MC 6800 : Minimum cycle time = 1.0×10^{-6} sec.
Processing time = $(1206)(1.0 \times 10^{-6}) = 1.2$ milliseconds
$>$ 0.833 milliseconds.

4) Z-80. The minimum cycle time = 4.0×10^{-6} seconds.
Hence, this also will take more than 833 milliseconds.

Example 4 : Microprocessor based spectral analysis for real time applications

The major areas in which spectral analysis is extensively used are Electronic and Communication Engineering, Mechanical Vibrations and structural analysis. This is used to detect, identify and characterise the natural and man made processes which change in time or space.

Suppose a sound spectra of a person has been recorded, then this analysis could be used to find whether a new sound belongs to the same person or not. The method used for this purpose is the DFT (Discrete fourier transform). It is discrete

time and discrete frequency representation of the fourier transforms. DFT could be calculated by FFT (in the example 1 for calculating FFT algorithm used was Cooley-Tukey). Here the method used is Recursive algorithm or 'Recursive way of computing DFT'. For more details please refer to[SES 80]. The time required for processing is again dependent upon the multiplication time. Number of multiplications to be carried out by this algorithm = M^2 (where M is the number of points). Here like in example 1, the process should be completed within 620 milliseconds.

Agains, like in example 1, the number of points considered is 1024 for optimum result.

One complex multiplication requires 4 real multiplication and two real additions.

Hence, the total number of multiplications = $(1024)^2 x4$

For 16x16 bits multiplication it is = $4x4x(1024)^2$ = 1048576.

[Assuming that 16x16 bits multiplication takes about 4 times the time taken for 8x8 bits multiplication].

MC 6800 takes about 56 clock cycles for multiplication.

Therefore, total number of clock cycles required for multiplication = 1048576x56x8

= 58720000.0.

1) Intel 8086

Minimum cycle time = 0.2×10^{-6} seconds.

Time required for multiplication = $4 \times (1024)^2 \times 136 \times 0.2 \times 10^{-6}$ sec

[In this case for 16x16 bits multiplication takes 136 cycles] = 7.13 seconds $>>$ 620 milliseconds.

2) Intel 8085

Minimum cycle time = 0.32×10^{-6} seconds.

Time required for multiplication = $58720000.0 \times (0.32) \times 10^{-6}$

= 18.79 seconds $>>$ 7620 milliseconds.

3) MC 6800

Minimum cycle time = 1.0×10^{-6} seconds.

For multiplication it will take much more time than 8085.

4) Z-80

Minimum cycle time = 4.0×10^{-6}. This also will take much more time for multiplication compared to 8085.

Here the point to be emphasised is of processing. The critical points are :

1) Cycle time of microprocessor : Should be less than 0.0105 μsec.

2) Processing cycles : This is the number of operations to be performed.

3) Sampling rate.

A graph [Fig. 4.2] has been drawn indicating the processing time vs the number of points.

Example 5 : Air Data Computer

It consists of temperature and pressure sensors. These converts the parameters into electrical signals. The air data computer (ADC) computes the various parameters required by the pilot.

The three inputs are :

1) Static pressure : This is the absolute pressure of the still air surrounding the aircraft. The pressure is usually sensed by providing a port on the aircraft skin.
2) Dynamic pressure : It is picked up by pitot tube.
3) Air temperature : For ambient temperature measurement, the probe is to be positioned on the exterior of an aircraft. This probe gives the total temperature and from this static temperature could be calculated.

The outputs given by ADC are :

1) Altitude, 2) Indicated air speed (IAS), 3) True air speed (TAS) and (4) Mach number.

These outputs are to be updated every second. The inputs are to be sampled every 100 milliseconds. This has been found satisfactory.

The time required to carry out the calculation as indicated in[SOV 78]are :

1) Altitude takes about 7.0 milliseconds

2) IAS takes about 9.0 milliseconds

3) Mach number takes about 10.0 milliseconds

4) TAS takes about 12.0 milliseconds.

The total number of processing cycles required = 136000.00 [SOV 78]. Here, neither the processing time nor the sampling frequencies are critical.

Factors on which air data computer could be designed

1) Cycle time of microprocessor : Should be $\leq$ 0.735 μsec.

2) Processing cycles : Number of operations to be performed, and is equal to 136000.00

3) Sampling frequencies.

Individually, let us look at the microprocessors.

1) Intel 8086

Minimum cycle time = 0.2×10^{-6} second

Processing time = $136000.00 \times 0.2 \times 10^{-6}$ = 27.2 milliseconds $<$ 100 milliseconds.

2) Intel 8085

Minimum clock cycle = 0.32×10^{-6} seconds.

Processing time = $136000.0 \times 0.32 \times 10^{-6}$ seconds.

= 43.52 milliseconds $<$ 100 milliseconds.

3) MC 6800

Minimum clock cycle = 1.0×10^{-6} seconds

Processing time = $136000.00 \times 1.0 \times 10^{-6}$ seconds

= 136 milliseconds $\geq$ 100 milliseconds.

4) Z-80

Minimum clock cycle of Z-80 is greater than that of MC 6800. Therefore, Z-80 also cannot be utilized for ADC.

Here, except Z-80 and MC 6800 all the microprocessors could be utilised.

Example 6 : A Microprocessor Based Electronic Exchange

Here we try to study the feasibility of a microprocessor in Telephone exchange. The microprocessors that are discussed here are 8085, 6800, Z-80 and 8086. There are two approaches possible :

1) Scanning the telephone lines : Here the microprocessor scans the telephone lines periodically and based on this scanning, connects or disconnects the telephone lines of the user. If the number of telephones are more, then the time for scanning is more and thus it is difficult to keep the necessary connection between telephones. Therefore, with this method number of telephones that can be serviced is quite low. Further, the junctions cannot be scanned by this process. For this, some additional hardware is required viz. circulating memory etc. For more details please refer to[KOH 78].

2) Scanning the junctions : The interrupt generated by the lines give rise to scanning. With this method the number of telephones that can be serviced is large compared to the first method.

In this section we have assumed the following simplification.

i) Junctions are not scanned directly. Each junction is assigned a permanent memory location, and only these locations are used for scanning purposes. Since direct bit manipulation is not

possible, it is preferred to have a byte for each junction [VIZ 8085]. If we have to do the bit manipulation indirectly than the number of instructions will be more. It is seen that, due to this data structure the total memory used will be more. But, for optimum usage, we think it is better to use more memory location rather than waste time for processing of bits.

This array is called Junction. The status of each function can be obtained from this array.

ii) An array called Engaged, which keeps track of the status of each telephone. Here again each telephone has been assigned a byte in the Engaged array for the same reason as in (i).

iii) A third array is utilised to write the junction number corresponding to the telephones, i.e. when a telephone is assigned a junction, that junction number is written corresponding to that telephone number. This array is called Available.

iv) Further it is being assumed that each telephone has associated hardware which gives out status etc. as shown below :

i) Calling party ⟨Called party Number, status⟩

Status = 0 → Calling party

Status = 1 → Called party

Status = 2 → Over.

ii) Called party ⟨its number, status⟩

iii) Over i.e. keeps the receiver down ⟨its number, status⟩

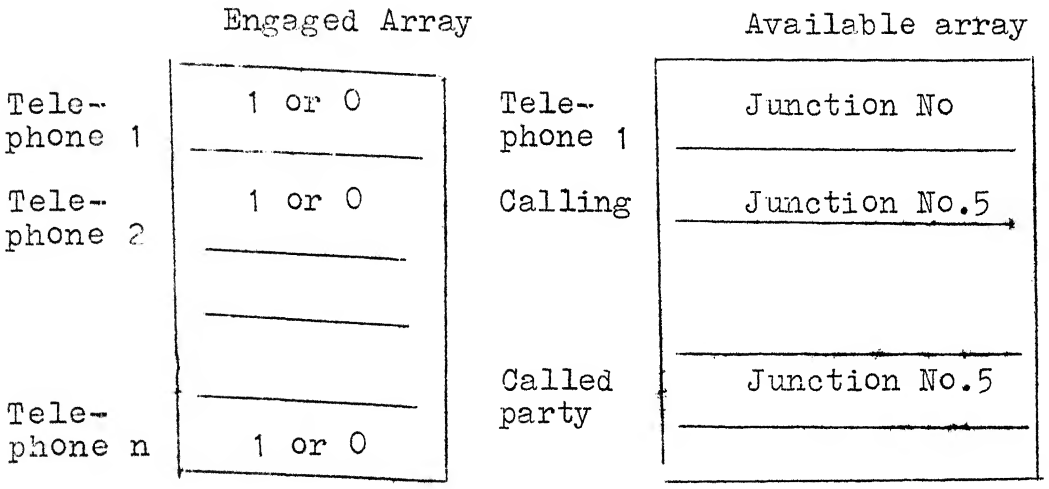
iv) A calling party has to wait for a maximum duration of three seconds for this call to be processed.

v) 10 percent of the calls are at any instant are made.

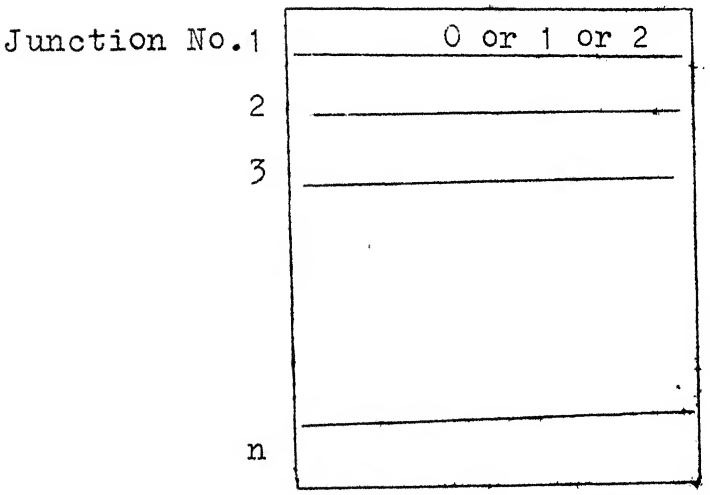
The program flow is shown below :

```
Case status of :
0 : If engaged [called party line number ] then
    Return engaged tone to the calling party
    Else
       Begin
          Search for Junction [Index] ;
             Junction [Index] : = 1;
             Engaged [Calling party] : = 1;
             Engaged [Called party]: = 1;
             Available [Calling party] : = Index of Junction;
             Send signal to called party;
       end;
1 : Junction [Index]: = Junction [Index] + 1;
2 : Engaged [Telephone No]: = 0;
    Junction [Index]: = Junction [Index] - 1;
```

Data Structure



Junction array



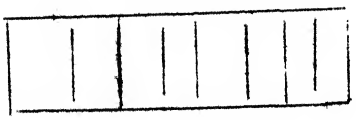
0 → Free

1 → Calling party has been alloted the junction

2 → Called party has also been alloted the same junction

Port gives the Data in the following manner

8 bits called party (lower byte)



Highest two bits for status and reamining 6 bits for called party

The program for the telephone exchange has been written in Appendix B.

From the program it is found out that the processing cycles = 604 + 56 X (No. of junctions).

Following graphs have been drawn.

1) Processing time vs number of junctions for all micro-processors [Fig. 4.3]
2) Memory locations vs number of junctions [Fig. 4.4]
3) Number of telephones vs number of junctions [Fig. 4.5]

The emphasis is on sampling. The sampling, directly affects the processing time, since sampling is done inside the processor memory, where corresponding to each junction one memory space has been allocated.

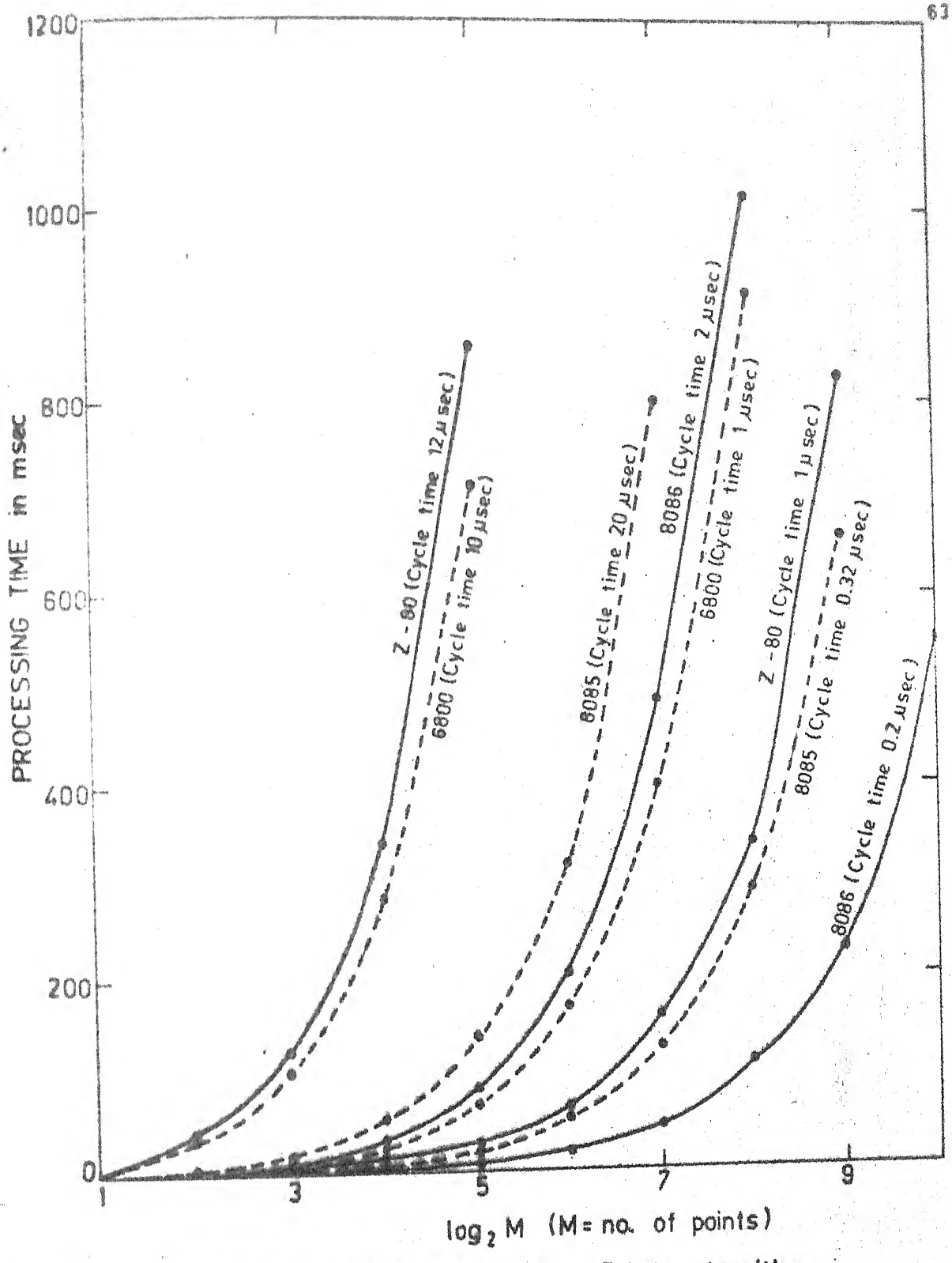


Fig. 4.1. FFT using Colley - Tukey algorithm.

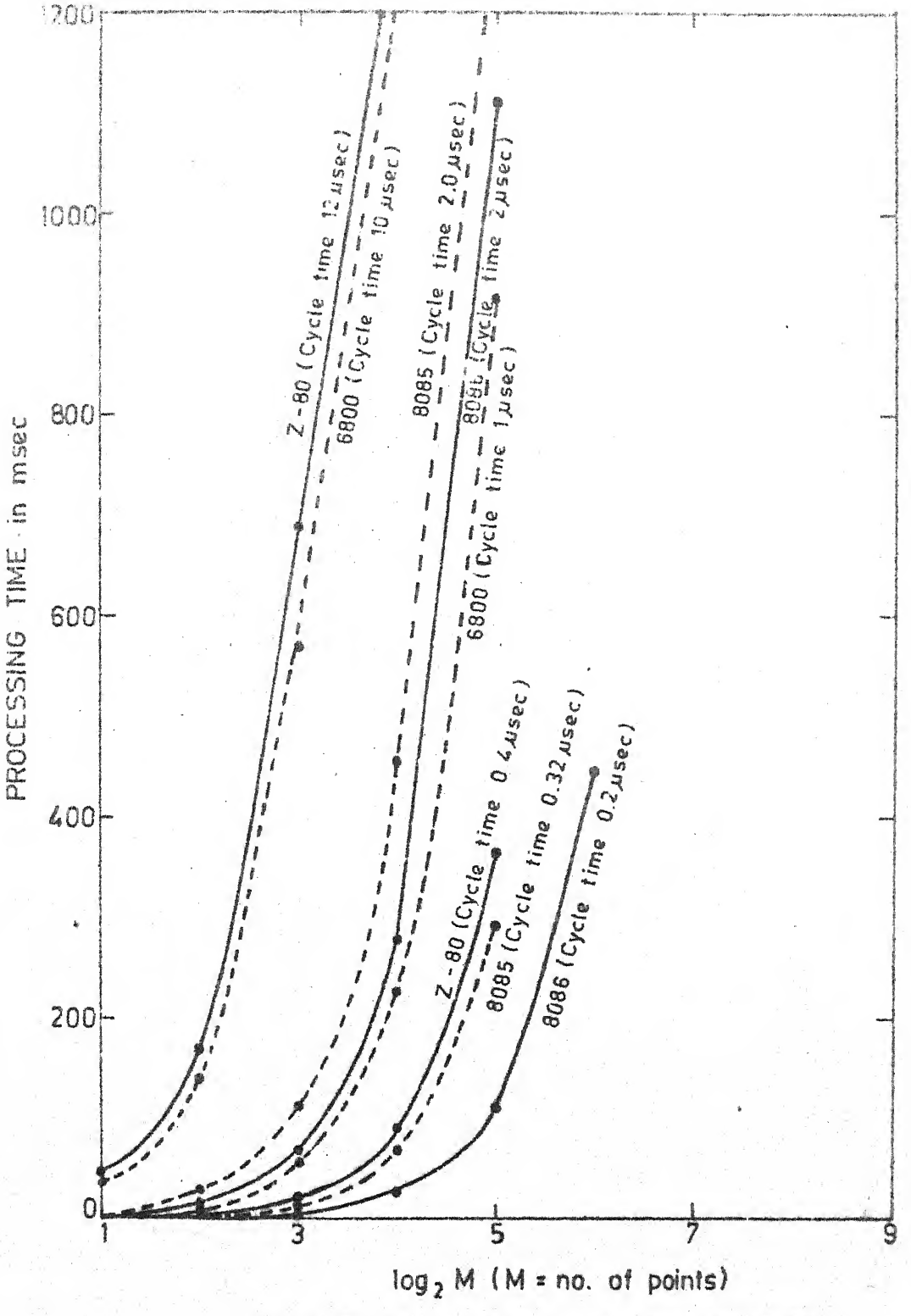


Fig. 4.2. Recursive way of computing DFT.

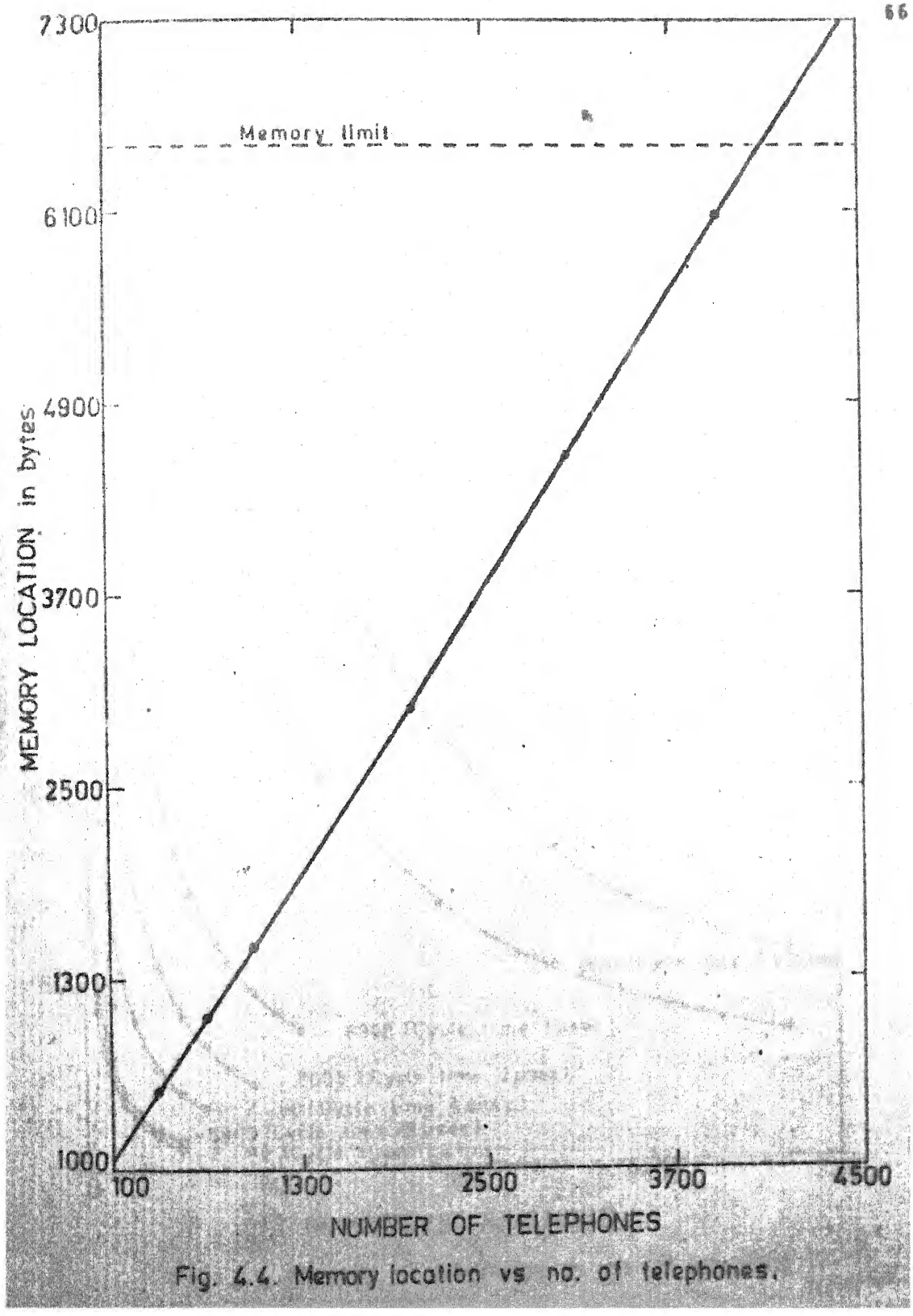


Fig. 4.4. Memory location vs no. of telephones.

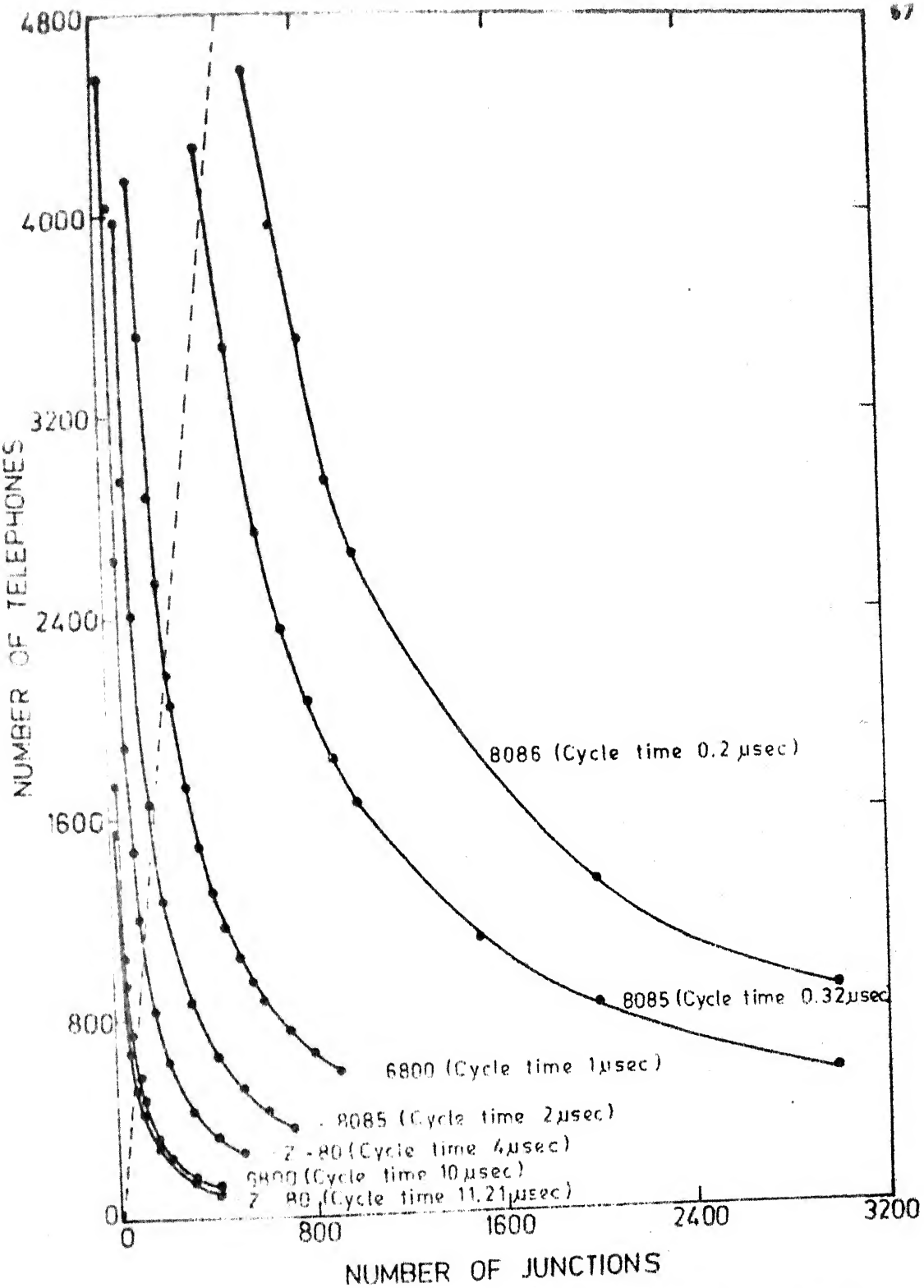


Fig 4.5. Number of telephones vs no. of junctions.

CHAPTER V

SELECTION ALGORITHM

In Chapter III, we have characterised a microprocessor based on its various properties. These parameters of the microprocessor will be embedded in the selection algorithm, which will be the subject of this chapter.

In Chapter IV, a number of applications have been characterised. Various applications have been considered and the parameters which are critical have been mentioned.

Given an input characterising an application, [the characteristics of the various microprocessors (Intel 8085, MC 6800, Zilog Z-80 and Intel 8086) are embedded in the selection algorithm], the selection algorithm comes up with a microprocessors, which is best suited for the given application. Selection algorithm is an important aid to an applications engineer, since he does not have to go through all the microprocessors to find out the best fit for his data inputs. He only has to characterize his application in terms of processing cycles and input sampling rate.

5.1 SELECTION ALGORITHM

Selection algorithm takes into account the following parameters.

1. Cycle time of the microprocessors : As has been mentioned in Chapter III, the main pivot of the selection algorithm is the cycle time because if the cycle time is low then the instruction execution time is also low (Instruction execution covers both processing and I/O handling). Therefore, it goes without saying that, if the cycle time is large then the processing time is also large. Mathematically it means that instruction execution time (this time covers both processing and I/O handling) is proportional to cycle time.

2. A particular microprocessor can be operated for a particular bandwidth of cycle time. Within this time, one particular microprocessor may be the best for one type of I/O handling. When we say type of I/O handling it means either software polling or vector interrupt approach or DMA approach.

3. Processing time : If the user has an idea about the processing cycles required for this data input, then it will also be taken care by this algorithm. Processing cycle here means, the cycles required for actual calculation to be carried out inside the processor.

4. Data inputs : This has been dealt with in Chapter IV. Here the algorithm takes the given data by the user for his application viz. number of channels to be serviced, frequency with which these channels have to be serviced.

Processing by the selection algorithm :

Case (i) : The user gives the following type of information to the selection algorithm.

Processing cycle = Given, No. of channels = given, sampling frequency = given. Sampling method = given. The cycle time of the microprocessor will be computed by this algorithm in order to meet the value above input parameters.

Since the cycle time is not given, the selection algorithm tries to match all the microprocessors with the given input for the given sampling method. If it is not possible to match the given inputs by the given sampling method, then the selection algorithm tries to match the combination of this given sampling method and the next sampling method for all the microprocessors. If this approach also does not match the given inputs, then the selection algorithm tries to match the next sampling method alone for all the microprocessors. This way the selection algorithm tries to match the given inputs with that of the microprocessors for various sampling methods and gives an output, which tells the user that for different cycle times which microprocessor should be used and which sampling method has to be incorporated. The selection algorithm also gives the interface unit number for the sampling method that it has found suitable.

Case (ii) Cycle time = given . processing cycle = given No. of channels = given , sampling frequency = given sampling method = given.

In this case the selection algorithm checks for the microprocessors which can operate in the given cycle time. After getting the microprocessor, it tries to match the given inputs with the characteristics of the microprocessors for that sampling method. If by the given sampling method the selection algorithm cannot match the given inputs with that of the microprocessor, then it tries the combination of this sampling method and the next sampling method, until the inputs are matched with the characteristics of the microprocessor. If the selection algorithm cannot match the inputs at all, then it gives an output to that effect otherwise it indicates to the user the name of the microprocessor and its interface unit to be used for the given inputs.

Case (iii) Cycle time = not given. processing cycle = not given, no. of channels = given, sampling frequency = given, sampling method = given.

Here, since the processing cycle has not been given, the selection algorithm takes a value which is already present in it, and proceeds as in case (i).

Structure of the selection algorithm :

The structure is given in Pascal like language below and and the program listing is given in Appendix C

Main program

Begin

Call write messages;

Assume that four microprocessors are to be checked;

Assume software polling;

Read cycle time;

IF CTIME $\neq$ 0.0 then

Begin

Choose one microprocessor;

Call FINDMICRO;

end;

Read processing cycles;

Read sampling method;

Based on the input the method of sampling is chosen;

Read number of channels;

Read sampling frequencies;

Put the sampling frequencies in an array;

Calculate total number of samples;

While (No. of microprocessors $>$ 0) do

Begin

Call process for;

After checking the first micro go to the second micro;

end.

```
Procedure process for
Begin
   Procedure write result
      Begin
         Microprocessor number and associated cycle time;
         The type of I/O handling by which the inputs have been
         matched;
         The interface unit number;
         The number of channels which have to be sampled;
      end;
end.

Procedure compute
Begin
   Calculate which method of sampling could be used;
   Calculate which are the channels that can be serviced by
   different methods;
end.
```

```
Procedure sort
Begin
   Procedure interchange
      Begin
         Interchange the position of the channels;
      end;
   Arrange the PRF [I] in descending order of magnitude;
end.

Procedure find micro
Begin
   Based on the cycle time select the microprocessor;
   If cycle time is not given then select all the four
   microprocessor;
end.
```

5.2 EXPERIMENTAL RESULTS

Legend :

PC = processing cycles, NC = Number of channels

CT = cycle time in μsecs. SF = sampling frequency in Hz.

Application 1 : Fast fourier transforms using Colley-Tukey Algorithm

S.No.	Input	Output	Results/Discussions
1.	PC = 36700160 NC = 1 CT = 0.0 SF = 1.60	None of the microprocessors can be used	Theoretically, it has been indicated in Chapter IV that none of the microprocessors can be used for this application which is consistent with the output of the selection algorithm
2.	PC = 2867200 NC = 1 CT = 0.20 SF = 1.60	INTEL 8086 Software polling interface unit: 8282	Intel 8086 can be used for the given purpose and the method of I/O handling is software polling. Further the interface unit to be used is 8282. Theoretically similar results have been obtained as indicated in Chapter IV.

Application 2 : Microprocessor in fire control application

S.No.	Input	Output	Results/Discussions
1.	PC = 135370 NC = 1 CT = 0.0 SF = 10	INTEL 8086 Software polling Interface unit 8282 INTEL 8085 Software polling Interface unit 8225	Here the selection algorithm, checks for all the microprocessors and comes with the result as indicated. Theoretically also the same has been confirmed in Chapter IV.
2.	PC = 135370 NC = 1 CT = 0.20 SF = 10	INTEL 8086 Software polling Interface unit 8282	Since the cycle time is given, the selection algorithm, checks for the suitable microprocessor whereas in the previous instance it checked for all the microprocessors.

S.No.	Input	Output	Results/Discussion
3.	PC = 135370 NC = 1 CT = 0.32 SF = 10	INTEL 8085 Software polling Interface unit 8225	Again the selection algorithm checks for suitable microprocessor.
4.	PC = 135370 NC = 1 CT = 1.0 SF = 10	Motorola 6800 cannot be used	With the given cycle time the best microprocessor is 6800 but it cannot be used. Theoretically the same has been confirmed in Chapter IV.

Application 3 : CRC checking in Protocol

S.No.	Input	Output	Results/Discussions
1.	PC = 1206 NC = 1 CT = 0.0 SF = 1200	INTEL 8086 cannot be used INTEL 8085 DMA approach Interface unit 8257 DMA	8086 cannot be used since the interface unit has not been marketed yet. The other results satisfy the theoretical calculation of Chapter 4.

S.No.	Input	Output	Results/Discussions
2.	PC = 1206 NC = 1 CT = 0.32 SF = 1200	INTEL 8085 DMA approach Interface unit 8257 DMA	Here the cycle time is given hence the selection algorithm checks for a suitable microprocessor to satisfy the given cycle time and the given inputs.

Application 4 : Microprocessor based spectral analysis for real time applications

S.No.	Input	Output	Results/Discussions
1.	PC = 5.8720E+07 NC = 1 CT = 0.0 SF = 1.6	None of them can be used	Since the cycle time is not given, the selection algorithm checks for all the microprocessors and finds that none of them can be used. This is confirmed by the theoretical calculation of Chapter 4.

Application 5 : Air data computer

S.No.	Input	Output	Results/Discussions
1.	PC = 136000 NC = 7 CT = 0.0 SF = 0.1,0.1, 0.1,1.0, 1.0,1.0, 1.0	INTEL 8086 Software polling Interface unit 8282 INTEL 8085 Software polling Interface unit 8225 Motorola 6800 Software polling Interface unit 6820	Since the cycle is not given, the selection algorithm checks for all the microprocessors and indicates to the user that except Z-80, all the microprocessors can be used.

Application 6 : Electronic Telephone Exchange

S.No.	Input	Output	Results/Discussions
1.	PC = 1164 NC = 1 CT = 0.0 SF = 10	INTEL 8086 Vector interrupt method, Interface unit 8288 INTEL 8085 Vector interrupt method, Interface unit 8259 Motorola 6800 Vector interrupt method, Interface unit 6828 Zilog Z-80 Vector interrupt method, Interface unit Z-80 PIO	Since the cycle time is not given, the selection algorithm checks for all the microprocessors and indicates that for ten junctions all the microprocessors could be utilized.

S.No.	Input	Output	Results/Discussions
2.	PC = 2564 NC = 1 CT = 0.0 SF = 35	INTEL 8086 Vector interrupt method, Interface unit 8288 INTEL 8085 Vector interrupt method, Interface unit 8259 Motorola 6800 Vector interrupt method Interface unit 6828 Zilog Z-80 Vector interrupt method, Interface unit Z-80 PIO	Here also 35 junctions can be serviced by all the microprocessors.

S.No.	Input	Output	Results/Discussions
3.	PC = 2844 NC = 1 CT = 0.0 SF = 40	INTEL 8086 Vector interrupt method, Interface unit 8288 INTEL 8085 Vector interrupt method, Interface unit 8259 Motorola 6800 Vector interrupt method, Interface unit 6828	Except Zilog Z-80,all the microprocessors can service 40 junctions. Theoretically it is seen that Zilog Z-80 can service 60 junctions. This descrepency is due to the fact that Z-80 is very slow and in the theoretical calculation the I/O handling has not been taken into account, whereas the selection algorithm takes everything into account.Hence, 35 is to be taken as the correct result.

S.No.	Input	Output	Results/Discussions
4.	PC = 7604 NC = 1 CT = 0.0 SF = 125	INTEL 8086 Vector interrupt method, Interface unit 8288 INTEL 8085 Vector interrupt method, Interface unit 8259 Motorola 6800 Vector interrupt method, Interface unit 6828	According to the selection algorithm, 125 junctions can be serviced by the 3 microprocessors mentioned.

S.No.	Input	Output	Results/Discussions
5.	PC = 7884 NC = 1 CT = 0.0 SF = 130	INTEL 8086 Vector interrupt method, Interface unit 8288 INTEL 8085 Vector interrupt method, Interface unit 8259	According to selection algorithm 8086 and 8085, can service 130 junctions. MC 6800 cannot service 130 junctions. Therefore, the result taken is that MC 6800 can service 125 junctions. Theoretically also similar results are obtained.
6.	PC = 13484 NC = 1 CT = 0.0 SF = 230	INTEL 8086 Vector interrupt method, Interface unit 8288 INTEL 8085 DMA approach Interface unit 8257 DMA	According to selection algorithm, 8086 could be used. 8085 can be used with DMA facility which is not possible in this problem. Hence, we will go back by 5 junctions, i.e. 225 junctions.

S.No.	Input	Output	Results/Discussions
7.	PC = 13204 NC = 1 CT = 0.0 SF = 225	INTEL 8086 Vector interrupt method, Interface unit 8288 INTEL 8085 Vector interrupt method, Interface unit 8259	Here we see that 225 junctions can be serviced by both 8086 and 8085 using vector interrupt approach. Hence, 8085 can service 225 junctions, which is very near the theoretical results (230 junctions).
8.	PC = 16844 NC = 1 CT = 0.0 SF = 290	INTEL 8086 Vector interrupt method, Interface unit 8288	Only 8086 can be used to service 290 junctions.
9.	PC = 17124 NC = 1 CT = 0.0 SF = 295	None of them can be used	Here 8086 also cannot be used. Hence, we take the previous value of 290 junctions. Theoretically also similar results are obtained.

From the results obtained from the selection algorithm it can be stated that, they are consistent with the theoretical and for some examples with the practical solutions.

CHAPTER VI

CONCLUSION

We have shown that a microprocessor can be characterised by its cycle time, clock cycles required for different I/O handling approach, availability of interface units and the speed with which it executes different instructions. These have been dealt with in Chapter III.

Similarly, a number of real world control applications are taken and each one of them have been characterised by the processing requirements, sampling frequencies and the number of channels that have to be sampled. This process of characterisation has been dealt with in Chapter IV.

Following is the summary of the work considered in this thesis.

1) Characterising of a class of commonly used microprocessors. The class includes Intel 8085, MC 6800, Zilog Z-80 and Intel 8086. The Intel 8086 is also in our study because it indicates the current trend towards higher word length.

2) A generalised model to characterize, a group of applications which are I/O bound.

3) An algorithm which would enable user to select a microprocessor to fit his application.

For the example on Fast Fourier transform using Colley-Tukey algorithm Intel 8086 has been found to meet all the requirements.

For the microprocessor in fire control application, Intel 8086 and 8085 have been found suitable.

For CRC checking in Protocols, Intel 8086 and 8085 have been found satisfactory.

In the case of microprocessor based spectral analysis for real time application, none of the microprocessors have been found suitable.

For the Air data computer example, Intel 8086 and 8085 have been found suitable.

For the Telephone Exchange problems, 35 junctions can be serviced by all the microprocessors. 125 junctions can be serviced by all the microprocessors except Z-80. 225 junctions can be serviced by Intel 8085 and 8086. Intel 8086 can service 290 junctions. The servicing is done by vector interrupt approach.

Suggestions for Further Work :

The selection algorithm does not take into account the cost of microprocessors as well as the psychological factor (familiarity, applicability, etc.) of the user. Also, the complexity of the actual hardware connections of a microprocessor to that of

interface unit and from interface units to the I/O device has also not been considered.

It would be prudent to say that, if the cycle time is not given by the user to the selection algorithm, then the selection algorithm chooses all the microprocessors that can be used for that particular application. Thus, the selection algorithm gives the user choices. Here, the user can use his familiarity factor for the final choice.

It would be worthwhile to create a data base for all the microprocessors available. Here, the parameters that are necessary for the selection algorithm, should be inserted in the data base.

A program linkage between the selection algorithm and the data base so generated could be designed. When a new microprocessor is introduced in the market, its relevant characteristics can be augmented in the data base.

BIBLIOGRAPHY

[ANA 77] : Anand Jagdish, 'Microprocessor application in fire control system', M.Tech. thesis, Dept of Electrical Engineering, I.I.T. Kanpur, 1977.

[BHA 78] : Bhakat, U.S., 'Microprocessor based system for computing fast fourier transforms', M.Tech. thesis, Dept of Electrical Engineering, IIT Kanpur, 1978.

[DAV 79] : Davis, H.A., 'Comparing architectures of three 16 bit microprocessors', Computer Design, July 1979, pp. 91-100.

[FAR 80] : Frrar, F.A., and Eidens, R.A., 'Microprocessor requirements for implementing control logic', IEEE Trans. on Automatic Control, vol. AC-25, No.3, June 1980, pp. 461-468.

[KOH 78] : Kohli Kewal, 'A microprocessor based electronic exchange', M.Tech. thesis, Dept of Electrical Engineering, IIT Kanpur, 1978.

[MAT 81] : Mathur, Aditya, 'Introduction to microprocessors', Tata McGraw-Hill, New Delhi, 1981.

[PEN 77] : Penney, B.K., 'The implications of microprocessor architecture on speed, programming and memory size', The radio and electronic engineer, vol. 47, No.11, November, 1977, pp. 522-528.

[PEA 77] : Peatman, B. John, 'Microprocessor based design', Kogakusha, International Student Edition, Japan, 1977.

[SES 80] : Seshagiri Rao, S.V., 'Microprocessor based spectral analysis for real time applications', M.Tech. thesis, Dept of Electrical Engineering, IIT Kanpur, 1980.

[SOV 78] : Sovakar, Tarun, 'Microprocessor based air data computer', M.Tech. thesis, Dept of Electrical Engineering, IIT Kanpur, 1978.

[WHI 75] : Whiting, J.S., 'An efficient software method for implementing polynomial error detection codes', Computer Design, vol. 14, No.3, March 1975, pp. 73-77.

APPENDIX A

CRC CHECKING IN PROTOCOLS

Program for CRC checking and associated functions are given below. This program has been written for 8085.

		Clock cycles		Clock cycles
CRC :	MVI A,-4	7	MOV E,A	4
	ADD HL	4	MVI HL,XXX	7
	MOV HL,A	4	XRA	7
	MOV B, (HL)	7	ADD SP,2	4
	MVI C,0	7	XTHL	16
	MVI D,0	7	MOV C, (HL)	7
	XRA C,D	7	XTHL	16
	MVI E,0	7	MVI E,0	7
	XTHL	16	ADD SP,2	7
	MOV E, (HL)	7	XTHL	16
	XTHL	16	MOV C, (HL)	7
	MOV A,HL	4	XTHL	16
	MOV XXX,A	7	MOV A,HL	7
	MVI HL,H table address	7	MVI HL,XXX	7
	MOV A, (HL)	7	MOV (HL), A	7
	MOV HL,E	4	MVI HL, L table location	7
	ADD A,(HL)	7	MOV A, (HL)	7

	Clock cycles
ADD A,C	7
MOV C,A	4
MOV D,E	4
MVI A, base address	7
ADD A, -6	7
JNZ NEXTCH	10
MOV A,HL	4
MVI HL,XXX	7
MOV, (HL), A	7
MVI HL,base address -2	7
MOV A, (HL)	7
MOV HL,A	7
MVI A,1	7
CMP (HL)	4
JZ EXIT 7	10
MOV HL,B	4
MOV A,D	4
CMP (HL)	7
JNZ SEQN 28	10
INC B	4
MOV HL,B	4
MOV A,C	4
CMP (HL)	7
JNZ SEQN 28	10
MVI HL,Base address -2	7
MOV A, (HL)	7
MOV (HL), A	7
MVI A,3	7
MOV (HL),A	7
JMP EXIT 8	10
SEQN28: MVI HL, base address -2	7
MOV A, (HL)	7
MOV HL,A	4
MVI A,0	7
MOV A, (HL)	7
JMP EXIT 8	10
EXIT 7: INR B	4
INR B	4
MOV HL, B	4
MOV A,D	4
MOV (HL), A	7
INR B	4
INR B	4
MOV HL,B	4
MOV A,C	4
MOV (HL), A	7

		Clock cycles
EXIT 8 :	XTHL	16
	MOV A,HL	4
	ADD A,6	4
	MOV HL,A	4
	MOV A, (HL)	7
	MOV XXX, A	7

Total processing cycles = 1206

This has been used in Chapter 5 by the selection algorithm.

APPENDIX B

PROGRAM FOR ELECTRONIC EXCHANGE

This program has been written for INTEL 8085.

		Clock cycles
TELE 1 :	MVI A,1	7
	STA ENGAGED 1	13
	MVI L, Lower byte of X	7
	MVI H, Higher byte of X	7
	MVI A, Lower byte of code for outport	7
	MOV M,A	7
	INR L	4
	MVI A, Higher byte of code for outport	7
	MOV M,A	7
	MVI L, Lower byte address of this telephone	7
	MVI H, Higher byte address of this telephone	7
	JMP ROUTO	10

		Clock cycles		Clock cycles
ROUTO :	IN PORT	10	ANI 3F	7
	MOV C,A	7	MOV B,A	7
	IN PORT	10	MVI A,40	7
	MOV B,A	7	ANA E	4
	ANI C0	7	JNZ STATUS 1	10
	MOV E,A	7	MVI A,80	7
	MOV A,B	7	ANA E	4
			JNZ STATUS 2	10

	Clock cycles
MOV A,L	7
MOV D,H	7
MVI L,Lower byte of some fixed location	7
MVI H, Higher byte of same fixed location	7
MOV M,A	7
MOV A,D	7
INR L	4
MOV M,A	7
MOV L,Lower byte of some fixed address	7
MOV H,Higher byte of same fixed address	7
MOV A,C	7
MOV M,A	7
INR L	4
MOV A,B	7
MOV M,A	7
MOV L,C	7
MOV H,B	7
MVI A,1	7
ANA M	4
JNZ BUSY	10

	Clock cycles
JNSEARCH:	
MVI H,Higher order address byte of the beginning of available array	7
MVI L,Lower -do-	7
MVI A,1	7
ANA M	4
JNZ JGENG	10
MVI A,2	7
ANA M	4
JNZ JGENG	10
MOV H,Higher byte address where the location of the calling party No.is kept	7
MOV L,Lower -do-	7
MOV A,M	7
MOV C,A	7
INR L	4
MOV A,M	7
MOV B,A	7
MOV L,C	7
MOV H,B	7
MOV A,E	7

	Clock cycles
MOV M,A	7
INR L	4
MOV A,D	7
MOV M,A	7
MOV D,H	7
MOV E,L	7
MOV H,Higher address byte of the location where the called party No.is kept	7
MOV L, Lower -do-	7
MOV A,M	7
MOV C,A	7
INR L	4
MOV A,M	7
MOV B,A	7
MOV L,C	7
MOV H,B	7
MOV A,E	7
MOV M,A	7
INR L	4
MOV A,D	7
MOV M,A	7
MVI A,1	7
DCR L	4

	Clock cycles
MOV M,A	7
MOV A,code for the tinckle	7
OUT port	10
RET	10
BUSY:MVI A,code for busy tone	7
X: OUT port (comes from TELE 1)	-
RET	10
JGENG: INR L	4
JMP INSEARCH	10
STATUS 1:MOV L,C	7
MOV H,B	7
MOV A,M	7
MOV L,A	7
MVI H,0	7
INR M	4
MVI L,Lower byte of the address where JUNCTION begins	7
MVI H,Higher -do-	7
COM: ANA M	4
JNZ XYZ	10
INR L	4
JMP COM	10

	Clock
XYZ : INR M	4
RET	10
STATUS 2 : MOV L,C	7
MOV H,B	7
MVI A,0	7
MOV M,A	7
MVI H, Higher order byte corresponding to this number from JUNCTION	7
MVI L, - do -	7
MOV A,M	7
MOV L,A	7
MOV H,0	7
DCR M	4
RET	10
Total No. of clock cycles	= 827

From the program it is found out that the processing cycles is equal to 604 + 56 X (No. of junctions). This has been used in Chapter 5 by the selection algorithm.

APPENDIX C

COMPUTER LISTING OF SELECTION ALGORITHM

```
label
        10;
const
    DEVNAMETYPe=packed array [1..7] of char;
var
    CYCLETIME:array[1..4] of real;
    INO:array[1..4,1..3] of integer;
    TP:array[1..3] of real;
    DEVNAME:array[1..4,1..3] of DEVNAMETYPe;
    CTIME,PTIME,STOT,ST1,XTIME,ST2,NEWPTIME:real;
    COUNT,I,MICNO,STARTPT,N,INDEX:integer;
    PRF:array[1..256] of real;
    SORTARRAY:array[1..256] of integer;
    CH:char;SAMPTYPE,SAMPMETHOD:integer;
procedure ERROR;
    begin
        WRITELN(TTY,'IMPROPER CYCLETIME,JOB ABORTED');
        goto 10
    end;
procedure INITPROC;
    begin   CYCLETIME[1]:=0.2E-6;
        CYCLETIME[2]:=0.32E-6;
        CYCLETIME[3]:=1.0E-6;
        CYCLETIME[4]:=10.0E-6;
        PTIME:=20.0E-6;
        INO[1,1]:=142;
        INO[1,2]:=120;
        INO[1,3]:=0;
        INO[2,1]:=142;
        INO[2,2]:=166;
        INO[2,3]:=84;
        INO[3,1]:=50;
        INO[3,2]:=37;
        INO[3,3]:=26;
        INO[4,1]:=189;
        INO[4,2]:=109;
        INO[4,3]:=145;
        DEVNAME[1,1]:='8282    ';
        DEVNAME[1,2]:='8288    ';
        DEVNAME[1,3]:='        ';
        DEVNAME[2,1]:='8225    ';
        DEVNAME[2,2]:='8259    ';
        DEVNAME[2,3]:='8257DMA';
        DEVNAME[3,1]:='6820    ';
        DEVNAME[3,2]:='6828    ';
        DEVNAME[3,3]:='6844    ';
        DEVNAME[4,1]:='Z-80PIO';
        DEVNAME[4,2]:='Z-80PIO';
        DEVNAME[4,3]:='Z-80PIO';
        SAMPMETHOD:=1;
    end;
```

```
procedure WRITEMESSAges;
      begin
          WRITELN(TTY,'0.2USEC AND 0.32USEC CYCLETIME USE 8086MICRO');
          WRITELN(TTY,'0.32USEC AND 1.0USEC CYCLETIME USE 8085MICRO');
          WRITELN(TTY,'1.0USEC AND 10.0USEC CYCLETIME USE 6800MICRO');
          WRITELN(TTY,'10USEC AND 11.21USEC CYCLETIME USE Z-80MICRO');
  WRITELN(TTY,'DEFAULT VALUE WILL BE ASSUMED IF YOU TYPE 0.0');
      end;
procedure FINDMICRO;
      begin
          if CTIME>=0.2E-6 then
            begin
                if CTIME<0.32E-6 then MICNO:=1 else
                if CTIME<1.0E-6 then MICNO:=2
                else
                if CTIME<10.0E-6 then MICNO:=3 else
                if CTIME<11.21E-6 then MICNO:=4
                else ERROR
            end
          else ERROR;
      end;
procedure SORT;
      var
          OVER:boolean;I:integer;
      procedure INTERCHANGE;
            var
                TEMP:integer;
            begin
                TEMP:=SORTARRAY[I+1];
                SORTARRAY[I+1]:=SORTARRAY[I];
                SORTARRAY[I]:=TEMP;
                OVER:=false
            end;
      begin
          for I:=1 to N do SORTARRAY[I]:=I;
          OVER:=false;
          while not OVER do
            begin
                OVER:=true;
                for I:=1 to N-1 do
                  begin
IF PRFCSORTARRAY[I]]<PRFCSORTARRAY[I+1]] THEN INTERCHANGE;
                  end;
            end;
      end;
```

```
procedure PROCESSFOR(MICNO,STARTPT:integer);
    var
        J:integer;FIRSTTIME:boolean;
        SAMPTYPE,SAMPMAXTYPe:integer;
        OVER:boolean;
      procedure WRITERESULt;
          var
              I,J,INDEX1:integer;
          begin
              case MICNO of
                    1:WRITELN('INTEL 8086 CYCLETIME=',CYCLETIME[1]);
                    2:WRITELN('INTEL 8085 CYCLETIME=',CYCLETIME[2]);
                3:WRITELN('MOTORALA6800CYCLETIME=',CYCLETIME[3]);
                    4:WRITELN('ZILOG Z-80 CYCLETIME=',CYCLETIME[4])
              end;
              for I:=1 to 10 do WRITE('-');
              WRITELN; I:=1; INDEX1:=INDEX;
              while I<=N do
                begin
                    case SAMPTYPE of
  1:WRITELN('SOFTWARE POLLING: ',DEVNAME[MICNO,1]);
  2:WRITELN('VECTOR INTERRUPT METHOD:',DEVNAME[MICNO,2]);
 3:WRITELN('DMA OR BLOCK TRANSEFER:',DEVNAME[MICNO,3])
                    end;
                    WRITE(' CHANNEL NOS');
                    for J:=I to INDEX1 do
                          WRITE(SORTARRAY[J]);
                    WRITELN;
                    I:=INDEX1+1;INDEX1:=N;SAMPTYPE:=SAMPTYPE-1
                end;
          end;
      procedure COMPUTE;
          var
              I:integer;
          begin
              ST2:=TRUNC((STOT*TP[SAMPTYPE-1]-1)/(TP[SAMPTYPE]))+1;
              ST1:=STOT-ST2; I:=0;
              while (ST2>0 ) and (I<N) do
                begin  I:=I+1; ST2:=ST2-PRF[SORTARRAY[I]];
                end;
              INDEX:=I

end;
    begin
  for J:=1 to 3 do TP[J]:=(INO[MICNO,J]+PTIME)*CYCLETIME[MICNO];
          FIRSTTIME:=true; SAMPTYPE:=STARTPT; OVER:=false;
          SAMPMAXTYPe:=3;
          if MICNO=1 then SAMPMAXTYPe:=2;
          while(SAMPTYPE<=SAMPMAXTYPe) and (not OVER) do
            begin
                if TP[SAMPTYPE]*STOT<=1 then OVER:=true; INDEX:=N;
                if OVER and (not FIRSTTIME) then   COMPUTE;
 if OVER then WRITERESULt; FIRSTTIME:=false;SAMPTYPE:=SAMPTYPE+1
          end;
```

```
FOR I:=1 TO N DO WRITE(PRF[I]:10:2,' ':5);
WRITELN(' CYCLES/SECOND');
FOR I:=1 TO 60 DO WRITE('-'); WRITELN;
WRITELN(' OUTFLOWS :');
      while (COLMT> 0 do
        begin
          PROCESSFOR(MICNO,STARTPT);MICNO:=MICNO+1;COUNT:=COUNT-1;
END;
FOR I:=1 TO 60 DO WRITE('-');WRITELN;
FOR I:=1 TO 60 DO WRITE('*');WRITELN;
FOR I:=1 TO 60 DO WRITE('-');WRITELN;
10:
  end.
```

EG-1901-M-BHA-MIC